Python for Statisticians

Iowa State University

April 29, 2017

Welcome!

What you'll need

- A text editor (preferably with syntax highlighting support for Python)
- Access to the command-line (use the "Terminal" app if you are using a Mac)
- A newer version of Python 2 (Python 2.7.13 is the latest, but even 2.7.6 should be okay) or Python 3 (such as Python 3.5.2)
- `pip`, the Python package manager (comes with most versions of Python)

Schedule

- Intro slides
- Learning the basics (data types, conditionals, functions, classes, map, reduce, filter)
- Tutorial: building a web app
- Tutorial: data wrangling with pandas

What is Python?

Figure 1: Python logo

- Interpreted programming language (like R, not like Java)
- General purpose (like Java, not like R)
- Dynamic type system (like R, not like Java)
- Supports object-oriented, imperative, and functional styles

NOTE: Python 2 vs Python 3

Most of the differences between Python 2 and Python 3 are subtle. Since Python 2 is not going away anytime soon, IMO it is better to learn Python 2 first and then pick up Python 3 when necessary.

Who uses Python?

Companies using Python

- Dropbox [a]
- Instagram (website)
- Disqus
- Google
- Venmo
- Quora
- YouTube
- Reddit
- Pinterest
- Yelp

[a] https://www.quora.com/Which-Internet-companies-use-Python

Who uses Python?

Types of programmers using Python

- Full-stack web developers
- Engineers building ML/AI-backed applications
- Statisticians / data scientists
- Researchers, especially in deep learning

How do you write Python code?

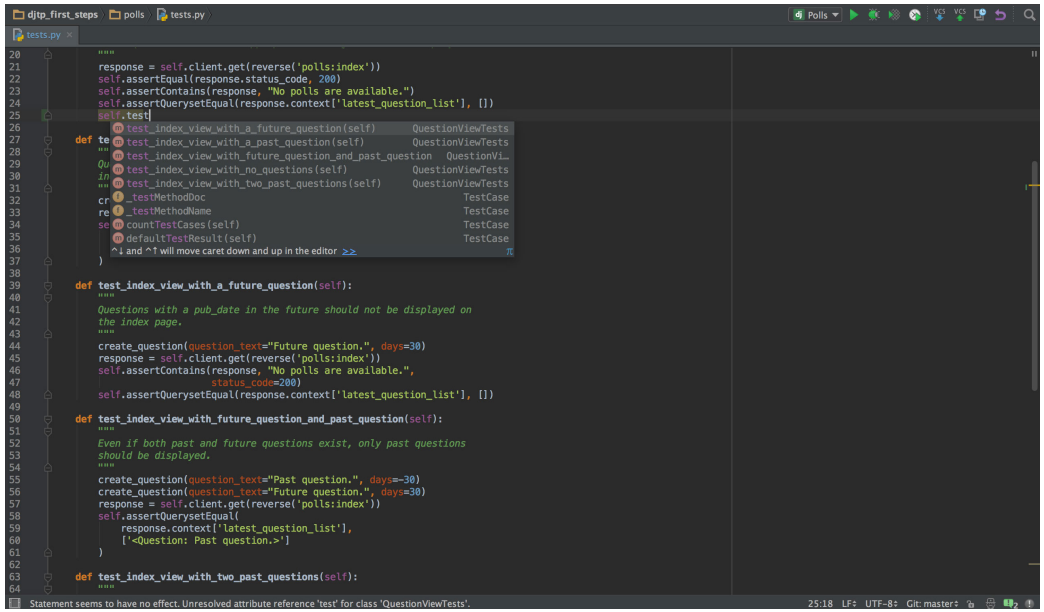

Figure 2: PyCharm IDE

Most people prefer to use an IDE (integrated development environment) such as PyCharm. However, all you really need is a basic text editor (preferably one with syntax highlighting support for Python) and access to the command line.

How do you write Python code?

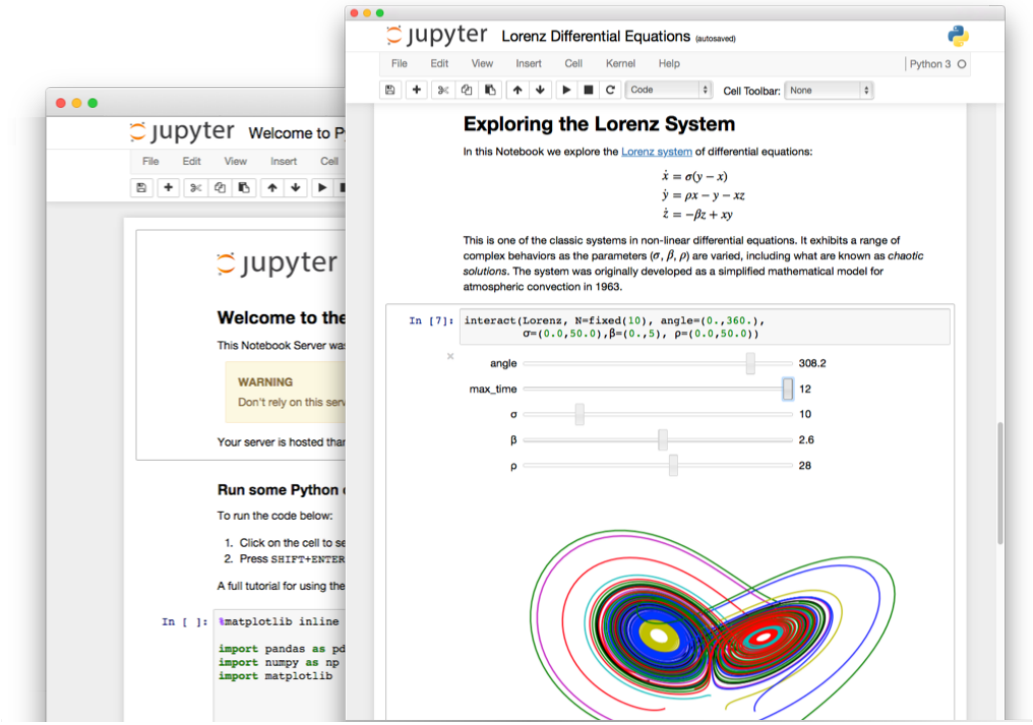

Figure 3: Jupyter Notebook

An alternate to a full-fledged IDE that can also be used to create runnable, interactive Python documents (sort of like RMarkdown), is the Jupyter Notebook.

Basic lingo

Script

A single file that is meant to define functions and/or classes or to be executed directly as a runnable program.

Module

A collection of functions and/or classes serving a particular (and somewhat narrow) purpose that are meant distributed.

Submodule

A sub-collection of functions and/or classes within a larger module.

Virtual environment

An isolated installation of Python that contains its own set of modules.

Syntax

- Spaces are used to separate logical blocks of code (unlike R, which uses brackets).
- Commented lines are preceded with #, and multi-line comments are placed between a triple set of quotes (single or double are both okay).

```
def hello_world():
    """
    This comment should describe what this function does.
    """
    return "Hello, World!"

for i in range(0, 10):
    # This is a comment
    if i > 5:  # also a comment, but this line is still ran
        print "Go away, World!"
    print hello_world()
```

Popular modules for data science

`numpy`

Provides array datatypes and efficient algorithms for sorting, slicing, and elementwise operations.

`scipy`

Built on top of `numpy`, `scipy` provides efficient algorithms for integration, optimization, parallel computing, etc.

`pandas`

Built on top of `numpy`, `pandas` provides a dataframe type of object along with fast data wrangling functions similar to what is provided by `dplyr` and `reshape2`.

Popular modules for data science

`sklearn`

Built on top of `numpy` and `scipy`, `sklearn` provides implementations for nearly every popular machine learning algorithm.

`matplotlib`

The core plotting functionality of Python is provided by `matplotlib`, analogous to the base-R plotting functions. There are many higher-level modules built on top of `matplotlib` that provide a more intuitive interface, such as `seaborn` and `plotly` (also an R package). There is even a version of `ggplot2` for Python.

`tensorflow`

Developed by Google, `tensorflow` is a popular open-source library for deep learning (actually, it's a lot more than that).

Style

Python code is beautiful. So let's keep it that way. The recommended styling guidelines for writing Python code are outlined in PEP 8 (Python Enhancement Proposal 8).

A few keys points:

- 4 spaces used to indent a block.
- Multi-line comment always used in function string (like in the `hello_world` example), preferably with double quotes.
- 2 spaces before function definition, except when defining class methods.
- Import statements should be placed at the top of the script right after the doc string, followed by two blank lines.